



22883

PATENT TRADEMARK OFFICE

103.1074.01

P01-1441

This application is submitted in the name of the following inventor(s):

<u>Inventor</u>	<u>Citizenship</u>	<u>Residence City and State</u>
Mark MUHLESTEIN	United States	Tucson, Arizona

The assignee is Network Appliance, Inc., a California corporation having an office at 495 East Java Drive, Sunnyvale, CA 94089.

Title of the Invention

Decentralized Virus Scanning for Stored Data

Background of the Invention

1. Field of the Invention

This invention relates to decentralized virus scanning for stored data, such as for example in a networked environment.

1 2. *Related Art*

2
3 Computer networking and the Internet in particular offer end users un-
4 precedented access to information of all types on a global basis. Access to information
5 can be as simple as connecting some type of computing device using a standard phone
6 line to a network. With the proliferation of wireless communication, users can now ac-
7 cess computer networks from practically anywhere.

8
9 Connectivity of this magnitude has magnified the impact of computer vi-
10 ruses. Viruses such as "Melissa" and "I love you" had a devastating impact on computer
11 systems worldwide. Costs for dealing with viruses are often measured in millions and
12 tens of millions of dollars. Recently it was shown that hand-held computing devices are
13 also susceptible to viruses.

14
15 Virus protection software can be very effective in dealing with viruses, and
16 virus protection software is widely available for general computing devices such as per-
17 sonal computers. There are, however, problems unique to specialized computing devices,
18 such as such as for example servers, file servers, storage systems, and devices of any kind
19 performing storage and retrieval of data. Off-the-shelf virus protection software will not
20 run on a specialized computing device unless it is modified to do so, and it can be very
21 expensive to rewrite software to work on another platform.

1 A first known method is to scan for viruses at the data source. When the
2 data is being provided by a specialized computing device the specialized computing de-
3 vice must be scanned. Device-specific virus protection software must be written in order
4 to scan the files on the device.

5
6 While this first known method is effective in scanning files for viruses, it
7 suffers from several drawbacks. First, a company with a specialized computing device
8 would have to dedicate considerable resources to creating virus protection software and
9 maintaining up-to-date data files that protect against new viruses as they emerge.

10
11 Additionally, although a manufacturer of a specialized computing device
12 could enlist the assistance of a company that creates mainstream virus protection software
13 to write the custom application and become a licensee this would create other problems,
14 such as reliance on the chosen vendor of the anti-virus software, compatibility issues
15 when hardware upgrades are effected, and a large financial expense.

16
17 A second known method for protecting against computer viruses is to have
18 the end user run anti-virus software on their client device. Anti-virus software packages
19 are offered by such companies as McAfee and Symantec. These programs are loaded
20 during the boot stage of a computer and work as a background job monitoring memory
21 and files as they are opened and saved.

1 While this second known method is effective at intercepting and protecting
2 the client device from infection, it suffers from several drawbacks. It places the burden
3 of detection at the last possible link in the chain. If for any reason the virus is not de-
4 tected prior to reaching the end user it is now at the computing device where it will do the
5 most damage (corrupting files and spreading to other computer users and systems).

6
7 It is much better to sanitize a file at the source from where it may be deliv-
8 ered to millions of end users rather than deliver the file and hope that the end user is pre-
9 pared to deal with the file in the event the file is infected. End users often have older ver-
10 sions of anti-virus software and/or have not updated the data files that ensure the software
11 is able to protect against newly discovered viruses, thus making detection at the point of
12 mass distribution even more critical.

13
14 Also, hand-held computing devices are susceptible to viruses, but they are
15 poorly equipped to handle them. Generally, hand-held computing devices have very lim-
16 ited memory resources compared to desktop systems. Dedicating a portion of these re-
17 sources to virus protection severely limits the ability of the hand-held device to perform
18 effectively. Reliable virus scanning at the information source is the most efficient and
19 effective method.

20
21 Protecting against viruses is a constant battle. New viruses are created eve-
22 ryday requiring virus protection software manufacturers to come up with new data files

1 (solution algorithms used by anti-virus applications). By providing protection at the
2 source of the file, viruses can be eliminated more efficiently and effectively.

3
4 Security of data in general is important. Equally important is the trust of the
5 end user. This comes from the reputation that precedes a company, and companies that
6 engage in web commerce often live and die by their reputation. Just like an end user
7 trusts that the credit card number they have just disclosed for a web-based sales transac-
8 tion is secure they want files they receive to be just as secure.

9
10 Accordingly, it would be desirable to provide a technique for scanning spe-
11 cialized computing devices for viruses and other malicious or unwanted content that may
12 need to be changed, deleted, or otherwise modified.

13 14 Summary of the Invention

15
16 The invention provides a method and system for performing specialized
17 services for files at a server, such as scanning files at a storage system, filer, or other
18 server performing storage and retrieval of data, for viruses by secondary computing de-
19 vices. The server (such as a filer) is connected to one or more supplementary computing
20 devices that scan requested files upon request to ensure they are virus free prior to deliv-
21 ery to end users. When an end user requests a file from the server the following steps oc-
22 cur: The server determines whether the file or other object requested by the user must be

1 scanned before delivery to, or after use by, the user. The server opens a channel to one of
2 the external computing devices and sends the filename (or some other designator of the
3 file or object, such as a file handle or an i-node pointer; "filename," "file name space"
4 and the like refer to the collection of possible designators for files or other types of ob-
5 ject). The external computing device opens the file and scans it. After possibly taking
6 remedial actions (such as for example cleaning the file of the virus, quarantining or de-
7 leting the file), the external computing device notifies the filer the status of the file scan
8 operation. The server sends the file to the end user provided the status indicates it may do
9 so.

10
11 This system is very efficient and effective, as a file needs only to be
12 scanned one time for a virus unless the file has been modified or new data files that pro-
13 tect against new viruses have been added. Scan reports for files that have been scanned
14 may be stored in one or more of the external computing devices, in one or more servers,
15 and some portion of a scan report may be delivered to end users.

16
17 In alternative embodiments of the invention one or more of the external
18 computing devices may be running other supplementary applications, such as data com-
19 pression and decompression, data encryption and decryption, and database compaction,
20 independently or in some combination.

1 Brief Description of the Drawings

2
3 Figure 1 shows a block diagram of a system for decentralized appliance vi-
4 rus scanning.

5
6 Figure 2 shows a process flow diagram for a system for decentralized virus
7 scanning

8
9 Detailed Description of the Preferred Embodiment

10
11 In the following description, a preferred embodiment of the invention is de-
12 scribed with regard to preferred process steps and data structures. Those skilled in the art
13 would recognize after perusal of this application that embodiments of the invention can
14 be implemented using one or more general purpose processors or special purpose proces-
15 sors or other circuits adapted to particular process steps and data structures described
16 herein, and that implementation of the process steps and data structures described herein
17 would not require undue experimentation or further invention.

1 *Lexicography*

2

3 The following terms refer or relate to aspects of the invention as described
4 below. The descriptions of general meanings of these terms are not intended to be limit-
5 ing, only illustrative.

6

7 ○ **filer** — In general, this refers to any storage system, file server, or other device per-
8 forming storage and retrieval of data. Storage systems might be implemented in any
9 one of a large variety of ways, including but not limited to a network-attached storage
10 environment; a storage area network; a disk assembly coupled to a client device, a
11 server device, or a host computer; or some combination thereof.

12

13 One type of storage system is a file server. A file server or filer includes a computer
14 that provides file services relating to the organization of information on writeable per-
15 sistent storage devices, such as memories, tapes or disks of an array. The filer might
16 include a storage operating system that implements a file system to logically organize
17 the information as a hierarchical structure of directories and files on, e.g., the disks.
18 Each "on-disk" file may be implemented as a set of data structures, e.g., disk blocks,
19 configured to store information, such as the actual data for the file. A directory, on
20 the other hand, might be implemented as a specially formatted file in which informa-
21 tion about other files and directories are stored. In general, the term "storage operating
22 system" refers to computer-executable code that implements data storage functional-

1 ity, such as file system semantics, and manages data access. A storage operating sys-
2 tem can be implemented as an application program operating over a general-purpose
3 operating system, such as UNIX® or Windows NT®, or as a general-purpose operat-
4 ing system with storage functionality or with configurable functionality that is config-
5 ured for storage applications, or as a special-purpose operating system dedicated to
6 performing a limited range of functionality including storage and related tasks in stor-
7 age appliances and other devices.

8
9 A storage system may be further configured to operate according to a client/server
10 model of information delivery to thereby allow many clients to access files stored on a
11 server, e.g., the storage system. In this model, the client may comprise an application
12 executing on a computer that "connects" to the storage system over a computer net-
13 work, such as a point-to-point link, shared local area network, wide area network or
14 virtual private network implemented over a public network, such as the Internet. Each
15 client may request the services of the file system on the storage system by issuing file
16 system protocol messages (in the form of packets) to the system over the network. It
17 should be noted, however, that the storage system may alternatively be configured to
18 operate as an assembly of storage devices that is directly-attached to a (e.g., client or
19 "host") computer. Here, a user may request the services of the file system to access
20 (i.e., read and/or write) data from/to the storage devices.

1 Although the invention is described herein with reference to a “filer,” there is no par-
2 ticular limitation of the invention to filers, file servers, storage systems, or similar de-
3 vices. It would be clear to those skilled in the art, after perusal of this application,
4 how to implement the ideas and techniques described herein for all types of server de-
5 vices. Such implementations would not require any undue experimentation or further
6 invention, and are within the scope and spirit of the invention.

7
8 ○ **i-node** — In general, this refers to a directory entry or other file descriptor entry per-
9 sistently maintained by a system performing storage and retrieval of data. In a pre-
10 ferred embodiment, each file has an i-node, and the i-node is persistently recorded in a
11 directory for that file. Although the term “i-node” is sometimes referred to in the
12 known art as being particular the Unix operating system and variants thereof, it is used
13 in this description much more generally, as noted herein. There is no particular re-
14 quirement in the invention that i-nodes must have any particular structure, or must be
15 stored in any particular format or place, or are specific to any particular operating
16 system, storage operating system, storage structure, hierarchical file system, file name
17 space, or storage paradigm.

18
19 ○ **file** or other **object** — In general, this refers to any data object at the server, whether a
20 sequential set of bytes, a set of records in a data base, a software object in an object-
21 oriented database or an object-oriented language development environment, or any
22 dynamically generated set of data for which a user request is appropriate. In a pre-

1 ferred embodiment, a file includes a set of data persistently recorded in a hierarchical
2 namespace and having a set of file attributes. While this is preferred, there is no par-
3 ticular requirement that a file or other object requested by the user have these proper-
4 ties, or any particular other properties, as the scope and spirit of the invention is broad
5 enough to include all types of objects.

6
7 ○ **virus** — In general, this refers to any manmade program or piece of code that is
8 loaded onto a computer without the computer user's knowledge and runs against their
9 wishes. Most viruses can also replicate themselves, and the more dangerous types of
10 viruses are capable of transmitting themselves across networks and bypassing security
11 systems. A "virus" can also include any malicious code, program, or other internal
12 component (including but not limited to a computer virus, computer worm, computer
13 time bomb, Trojan horse, or component with similar effect), that could damage, de-
14 stroy, alter, or take control of, software, firmware, or hardware, or could, in any man-
15 ner, reveal, damage, destroy, or alter any data or other information accessed through
16 or processed by the computer in any manner.

17
18 ○ **client and server** — in general, these terms refer to a relationship between two de-
19 vices, particularly to their relationship as client and server, not necessarily to any par-
20 ticular physical devices.

1 For example, but without limitation, a particular client device in a first relationship
2 with a first server device, can serve as a server device in a second relationship with a
3 second client device. In a preferred embodiment, there are generally a relatively small
4 number of server devices servicing a relatively larger number of client devices.

5
6 ○ **client device** and **server device** — in general, these terms refer to devices taking on
7 the role of a client device or a server device in a client-server relationship (such as an
8 HTTP web client and web server). There is no particular requirement that any client
9 devices or server devices must be individual physical devices. They can each be a
10 single device, a set of cooperating devices, a portion of a device, or some combination
11 thereof.

12
13 For example, but without limitation, the client device and the server device in a client-
14 server relation can actually be the same physical device, with a first set of software
15 elements serving to perform client functions and a second set of software elements
16 serving to perform server functions.

17
18 Although the invention is described with regard to a client-server model, there is no
19 particular requirement in the invention that the stored data is maintained and commu-
20 nicated to users using a client-server model. For example, other forms of distributed
21 computing in which a user request for access to data objects triggers decentralized

1 processing by one or more of a set of computing devices would also be within the
2 scope and spirit of the invention.

3
4 As noted above, these descriptions of general meanings of these terms are
5 not intended to be limiting, only illustrative. Other and further applications of the inven-
6 tion, including extensions of these terms and concepts, would be clear to those of ordinary
7 skill in the art after perusing this application. These other and further applications are
8 part of the scope and spirit of the invention, and would be clear to those of ordinary skill
9 in the art, without further invention or undue experimentation.

10
11 *System Elements*

12
13 Figure 1 shows a block diagram of a system for decentralized appliance vi-
14 rus scanning.

15
16 A system 100 includes a client device 110 associated with a user 111, a
17 communications network 120, a filer 130, and a processing cluster 140.

18
19 The client device 110 includes a processor, a main memory, and software
20 for executing instructions (not shown, but understood by one skilled in the art). Although
21 the client device 110 and filer 130 are shown as separate devices there is no requirement
22 that they be physically separate.

1
2 In a preferred embodiment, the communication network 120 includes the
3 Internet. In alternative embodiments, the communication network 120 may include alter-
4 native forms of communication, such as an intranet, extranet, virtual private network, di-
5 rect communication links, or some other combination or conjunction thereof.

6
7 A communications link 115 operates to couple the client device 110 to the
8 communications network 120.

9
10 The filer 130 includes a processor, a main memory, software for executing
11 instructions (not shown, but understood by one skilled in the art), and a mass storage 131.
12 Although the client device 110 and filer 130 are shown as separate devices there is no re-
13 quirement that they be separate devices. Moreover, although the invention is described
14 with regard to a single filer 130, the invention is equally applicable to sets of filers 130
15 operating with the processing cluster 140. A set of multiple filers 130 might each one op-
16 erate independently and each one make individual use of the processing cluster 140, or
17 might operate in conjunction as a group and make use of the processing cluster 140 as a
18 collective entity, or some combination thereof. Since, as noted below, the processing
19 cluster 140 can include one or more cluster devices 141, the invention can be performed
20 with any set of M filers and any set of N processors. There is no particular requirement
21 that M or N must be fixed; either filers 130 or cluster devices 141 might be added by op-

erator command or by a handshaking protocol while filers 130 and cluster devices 141 are operating. The filer 130 is connected to the communications network 120.

The filer 130 includes a set of configuration information 137 disposed so that a processor for the filer 130 can readily access that configuration information 137. In a preferred embodiment, the filer 130 includes software instructions for reviewing, reporting, editing, or modifying the configuration information 137, as directed by an operator, or possibly by a remote user having designated privileges. The configuration information 137 includes the following:

- Information indicating a first set of file types for which virus scanning is enabled (such as executable files, often designated by the file name extension EXE), and a second set of file types for which virus scanning is disabled (such as raw text files, often designated by the file name extension TXT);
- Information indicating a first file space for which virus scanning is enabled for all file operations (such as a first CIFS “share” designated by its root directory, for example /users/Swernofsky), a second file space for which virus scanning is enabled for file write operations only (such as a second CIFS “share”), and a third file space for which virus scanning is disabled (again, such as a third CIFS “share”);

1 and

- 2
- 3 ○ Information indicating for each file whether that file has been scanned for a virus,
4 and if so, what date and time that scan was performed (such as a timestamp), by
5 what type of scanning device or scanning software that scan was performed (such
6 as the make and version number of the scanning software), and what the results of
7 that scan were (such as whether a virus was detected and what actions were taken
8 if a virus was in fact detected). In a preferred embodiment, this information is re-
9 corded in an i-node for the file, or if the file is read-only or if the i-node is unwri-
10 table (such as if the file is part of a read-only snapshot), in a separate scanning
11 history database.
- 12

13 The mass storage 131 includes at least one file 133 that is capable of being
14 requested by a client device 110. The processing cluster 140 includes one or more cluster
15 device 141 each including a processor, a main memory, software for executing instruc-
16 tions, and a mass storage (not shown but understood by one skilled in the art). Although
17 the filer 130 and the processing cluster 140 are shown as separate devices there is no re-
18 quirement that they be separate devices.

19

20 In a preferred embodiment the processing cluster 140 is a plurality of per-
21 sonal computers in an interconnected cluster capable of intercommunication and direct
22 communication with the filer 130. There is no particular requirement that the processing

1 cluster 140 must be organized as a unified cluster, or must be local to the filer 130, or
2 must be homogeneous in the nature of the processing devices, or have any other particular
3 characteristics. For example, in alternative embodiments, the processing cluster 140 in-
4 cludes a set of PC's, workstations, servers, or other devices, coupled to the filer 130 by
5 means of a network such as the Internet.

6
7 In a preferred embodiment, cluster devices 141 in the processing cluster 140
8 register their presence with the filer 130, thus giving the filer 130 knowledge of their
9 availability to perform scanning (or other) operations. While this is preferred, there is no
10 particular requirement for the invention for registration, as the filer 130 may in alternative
11 embodiments be configured to send out "John Doe" requests for cluster devices 141 to
12 process files requested by the user.

13
14 The cluster link 135 operates to connect the processing cluster 140 to the
15 filer 130. The cluster link 135 may include non-uniform memory access (NUMA), or
16 communication via an intranet, extranet, virtual private network, direct communication
17 links, or some other combination or conjunction thereof.

18
19 *Method of Operation*

20
21 Figure 2 shows a process flow diagram for a system for decentralized appli-
22 ance virus scanning.

1
2 A method 200 includes a set of flow points and a set of steps. The system
3 100 performs the method 200. Although the method 200 is described serially, the steps of
4 the method 200 can be performed by separate elements in conjunction or in parallel,
5 whether asynchronously, in a pipelined manner, or otherwise. There is no particular re-
6 quirement that the method 200 be performed in the same order in which this description
7 lists the steps, except where so indicated.

8
9 At a flow point 210, the system 100 is ready to begin performing the
10 method 200.

11
12 At a step 211, a user 111 utilizes the client device 110 to initiate a request
13 for a file 133. The request is transmitted to the filer 130 via the communications network
14 120. In a preferred embodiment the filer 130 is an independent file server performing file
15 retrieval and storage in response to a file server protocol such as NFS or CIFS. In alter-
16 native embodiments, the filer 130 might be a supplemental storage device or file mainte-
17 nance server operating at the direction of another server, such as a web server.

18
19 At a step 212, the filer 130 receives the request for the file 133 and deter-
20 mines if the file 133 must be scanned for a virus. As part of this step, the filer 130 per-
21 forms the following sub-steps:

- 1 ○ At a sub-step 212(a), the filer 130 reviews its information regarding whether the
2 file 133 has already been scanned for a virus. In a preferred embodiment, that in-
3 formation includes whether a scan has already been performed, what date and time
4 that scan was performed (such as a timestamp), by what type of scanning device or
5 scanning software that scan was performed (such as the make and version number
6 of the scanning software), and what the results of that scan were (such as whether
7 a virus was detected and what actions were taken if a virus was in fact detected).
8 As noted above, in a preferred embodiment, this information is recorded in the i-
9 node for the file 133. If the file 133 has already been scanned and is marked avail-
10 able for use (and the filer determines that no re-scan is required), the filer 130
11 makes the file available to the user without performing the scanning operation.
- 12
- 13 ○ At a sub-step 212(b), the filer 130 reviews its information regarding what types of
14 files 133 it should scan for a virus. The filer reviews its configuration information
15 137 describing a set of file types (1) that should be scanned for a virus, such as ex-
16 ecutable files, macros, scripts, and the like, and (2) that should not be scanned for a
17 virus, such as raw text files and the like. This set of file types might be selected by
18 an operator for the filer 130, and is maintained with the configuration information
19 137. In a preferred embodiment, file types are identified by portions of the file
20 name for the file 133, such as a file name extension. Known file name extensions
21 include EXE for executable files and TXT for raw text files.
- 22

- 1 ○ At a sub-step 212(c), the filer 130 reviews its information regarding what file
2 spaces it should scan for a virus. The filer reviews its configuration information
3 137 describing which file spaces should be scanned for (1) all file operations, (2)
4 only file write operations, or (3) no file operations. Where the file space should be
5 scanned for all file operations, the filer 130 causes the file 133 to be scanned be-
6 fore the file 133 is opened for any read operation and after the file 133 is closed
7 after a write operation. Where the file space should be scanned for only file write
8 operations, the filer 130 causes the file 133 to be scanned after the file 133 is
9 closed after a write operation.

10
11 At a step 213, the filer 130, having determined that the file 133 should be
12 scanned, sends the file ID and path of the file 133 to the processing cluster 140 where it is
13 received by one of the cluster devices 141. As part of this step, the filer 130 performs the
14 following sub-steps:

- 15
16 ○ At a sub-step 213(a), the filer 130 sets a timer to a cluster processor timeout value,
17 indicating how long the filer 130 is willing to wait for a cluster device 141 to
18 work.
19
20 ○ At a sub-step 213(b), the filer 130 waits for the cluster device 141 to complete its
21 work. While doing so, the cluster device 141 (hopefully) performs step 215, step
22 217, and step 219 described below.

1
2 ○ At a sub-step 213(c), if the cluster device 141 responds before the timeout, the filer
3 130 proceeds with the step 219 below, using the results from the cluster device
4 141.

5
6 ○ At a sub-step 213(d), if the cluster device 141 does not respond before the timeout,
7 the filer 130 might proceed in one of two ways: (a) The filer 130 proceeds with the
8 step 219 below, acting as if the cluster device 141 refused user access to the file.
9 In this case, the filer 130 reports that the file is not available due to the scan having
10 failed. (b) The filer 130 sends an ARE-YOU-WORKING? message to the cluster
11 device 141. In this case, if the cluster device 141 responds, within a second but
12 shorter timeout, that it is still working on the file 133, the filer 130 returns to the
13 sub-step 213(b) and resets the timeout.

14
15 In a preferred embodiment, there is more than one cluster device 141, so the filer
16 130 can proceed to service requests for other files 133 even if the cluster device
17 141 scanning one particular file 133 takes a very long time. In alternative em-
18 bodiments, the filer 130 may reassign the scanning task to a second cluster device
19 141 if the filer 130 suspects that the first cluster device 141 has in fact crashed, be-
20 come unavailable, or otherwise is not likely to respond successfully with a virus
21 scan result for the file 133.

- 1 ○ In the event that the user making the original request for the file 133 gives up be-
2 fore the cluster device 141 reports on the file 133, the filer 130 still waits for the
3 report from the cluster device 141, and marks the file 133 with the results of the vi-
4 rus scan performed by the cluster device 141. Thus, if the cluster device 141 de-
5 termines that the file 133 has no virus (or alternatively, finds a virus but success-
6 fully removes it), the filer 130 marks the file as successfully scanned and available
7 for use. If the same user or a different user later requests the same file 133, the
8 filer 130 makes that file 133 available without a further scan, as described below.

9
10 At a step 215, the cluster device 141 uses the file ID and path to open the
11 file 133 in the mass storage 131 of the filer 130.

12
13 At a step 217, the cluster device 141 scans the file 133 for viruses. In a pre-
14 ferred embodiment, files are tasked to the processing cluster 140 in a round robin fashion.
15 In alternative embodiments files may be processed individually by a cluster device 141,
16 by multiple cluster device 141 simultaneously, or some combination thereof. Load bal-
17 ancing may be used to ensure maximum efficiency of processing within the processing
18 cluster 140.

19
20 In a preferred embodiment, the filer 130 groups cluster devices 141 into one
21 or more classes, such as primary and secondary, where all primary cluster devices 141 are
22 assigned, followed by secondary cluster devices 141. This allows an operator to direct

1 the filer 130 to use a first cluster device 141, such as for example available using a rela-
2 tively rapid connection, exclusively, but when the first cluster device 141 is unavailable
3 for any reason, to fall back to using a second designated cluster device 141, such as for
4 example available using a much less rapid connection.

5
6 There are several vendors offering virus protection software for personal
7 computers, thus the operator of the filer 130 may choose whatever product they would
8 like to use that supports the communication protocol with the filer 130 described herein.
9 They may even use combinations of vendors' products in the processing cluster 140,
10 when those combinations can operate using the communication protocol with the filer 130
11 described herein. In alternative embodiments, the filer 130 may operate with forms of
12 virus protection software that does not support the communication protocol with the filer
13 130 described herein, with some features (such as the timeout and ARE-YOU-
14 WORKING? message) not available to those forms of virus protection software. In fur-
15 ther alternative embodiments of the invention, continual scanning of every file 133 on the
16 filer 130 may take place.

17
18 The processing cluster 140 is highly scalable. The price of personal com-
19 puters is low compared to dedicated devices, such as filers, therefore this configuration is
20 very desirable. Additionally, a cluster configuration offers redundant systems availability
21 in case a cluster device 141 fails – failover and takeover is also possible within the proc-
22 essing cluster.

1
2 The cluster device 141 is assigned a special type of access (herein called
3 “OPEN-FOR-SCANNING”), so that the cluster device 141 can scan the file 133 regard-
4 less of whether it is already locked by another user. In a preferred embodiment, OPEN-
5 FOR-SCANNING mode is restricted to those devices the filer 130 can verify are actually
6 cluster devices 141. In a preferred embodiment, the filer 130 can restrict OPEN-FOR-
7 SCANNING mode to devices according to one or more of the following criteria:

- 8
- 9 ○ having one or more selected IP addresses;
 - 10
 - 11 ○ being included in one or more selected IP subnets;
 - 12
 - 13 ○ being included in one or more selected DNS domains;
 - 14
 - 15 ○ being accessible to the filer 130 via one or more selected physical interfaces;
 - 16
 - 17 ○ having a selected username or user privileges (such as “Administrator” or “Backup
18 Operator”) at the cluster device 141.
 - 19

20 In a preferred embodiment, OPEN-FOR-SCANNING mode access is re-
21 stricted to processes running as an NT “Service” on the cluster device 141. Thus, a se-
22 lected cluster device 141 might be in use by a user having no particularly special privi-

1 leges, while the cluster device 141 concurrently operates with a service running as "Ad-
2 ministrator" and thus being allowed by the filer 130 to have OPEN-FOR-SCANNING
3 mode access.
4

5 At a step 219, the cluster device 141 transmits a scan report to the filer 130.
6 The scan report primarily reports whether the file is safe to send. Further information
7 may be saved for statistical purposes (for example, how many files have been identified
8 as infected, was the virus software able to sanitize the file or was the file deleted) to a
9 database. The database may be consulted to determine whether the file 133 needs to be
10 scanned before delivery upon receipt of a subsequent request. If the file 133 has not
11 changed since it was last scanned and no additional virus data files have been added to
12 the processing cluster, the file 133 probably does not need to be scanned. This means the
13 file 133 can be delivered more quickly.
14

15 Other intermediary applications may also run separately, in conjunction
16 with other applications, or in some combination thereof within the processing cluster 140.
17 Compression and encryption utilities are some examples of these applications. These
18 types of applications, including virus scanning, can be very CPU intensive, thus
19 outsourcing can yield better performance by allowing a dedicated device like a filer to do
20 what it does best and farm out other tasks to the processing cluster 140.
21

As part of this step, the filer 130 might also perform the following sub-steps:

- At a sub-step 219(a), the filer 130 records information from the scan report in the i-node for the file 133, or in a separate scanning history database if the file is read-only, as noted above.
- In a preferred embodiment, the filer 130 includes software instructions for responding to an operator or a privileged remote user to reset the scanning information for a file. This allows an operator or a privileged remote user to force the filer 130 to rescan one or more selected files 133.

At a step 221, the filer 130 transmits or does not transmit the file 133 to the client 110 based on its availability as reported following the scan by the processing cluster 140. Some portion of the scan report may also be transmitted to the user. As part of this step, the filer 130 performs the following sub-steps:

- At a sub-step 221(a), if the report from the cluster device 141 indicates that the file 133 is unavailable due to being infected (and the file 133 was not disinfected by the cluster device 141), the filer 130 sends a message box to the requesting user giving at least some information from the report from the cluster device 141. The filer 130 can send this message box to a user making a CIFS request because the

1 CIFS protocol allows the filer 130 to know the IP (internet protocol) address for
2 the user. For NFS, the filer 130 would build a string indicating a path to the re-
3 quested file 133 and send a message to the user including that string.

4
5 At this step, a request for a file 133 has been received, the request has been
6 processed, and if possible a file 133 has been delivered. The process may be repeated at
7 step 211 for subsequent requests.

8
9 *Generality of the Invention*

10
11 The invention has wide applicability and generality to other aspects of proc-
12 essing requests for files.

13
14 The invention is applicable to one or more of, or some combination of, cir-
15 cumstances such as those involving:

- 16
17 ○ file compression and decompression — the cluster processors can be used to de-
18 compress data for delivery to users, and to compress data received from users for
19 storage.
20
21 ○ file encryption and decryption — the cluster processors can be used to decrypt data
22 for delivery to users, and to encrypt data received from users for storage.

1
2 ○ database compaction — the cluster processors can be used to compact data in a
3 database or other structured format for delivery to users, or to compact data re-
4 ceived from users for storage.

5
6 ○ general outsourcing of CPU intensive tasks from dedicated appliances to general
7 purpose computers — for one example, the cluster processors can be used to
8 translate between data stored in a first form into data presented to users in a second
9 form.

10
11 *Alternative Embodiments*

12
13 Although preferred embodiments are disclosed herein, many variations are
14 possible which remain within the concept, scope, and spirit of the invention, and these
15 variations would become clear to those skilled in the art after perusal of this application.